

AD-A054 975

MARYLAND UNIV COLLEGE PARK COMPUTER SCIENCE CENTER
CELLULAR GRAPH ACCEPTORS, 3.(U)

F/G 9/2

APR 78 A WU
TR-648

AFOSR-77-3271

UNCLASSIFIED

AFOSR-TR-78-1007

NL

| OF |

AD
A054975

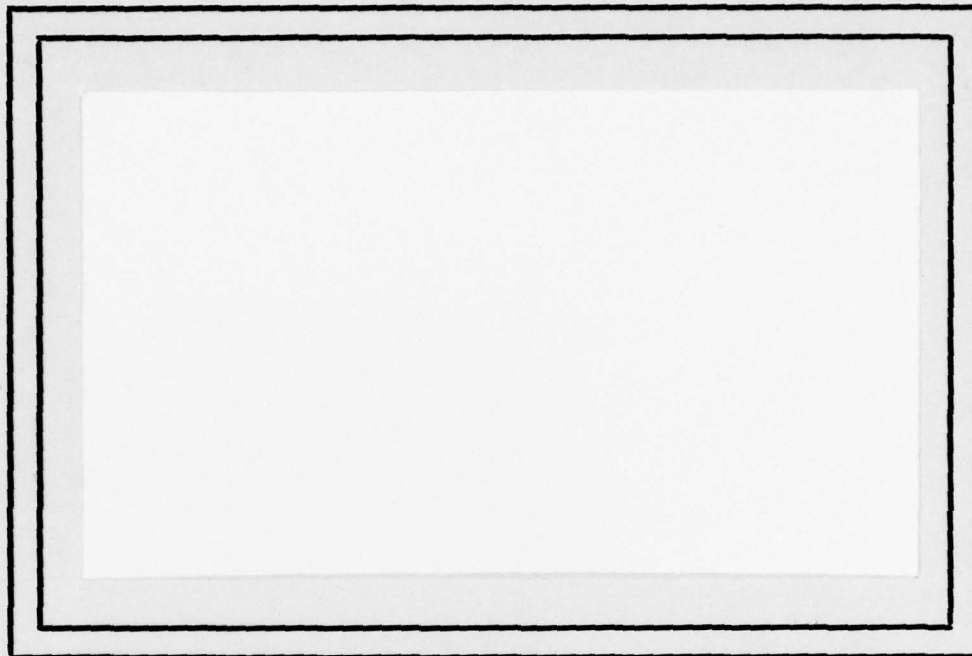


FOR FURTHER TRAN ~~MIT~~

AD-A050360 #2

2
with

AD A 054975



COMPUTER SCIENCE
TECHNICAL REPORT SERIES



DDC
RECEIVED
JUN 9 1978
B

UNIVERSITY OF MARYLAND
COLLEGE PARK, MARYLAND

20742

AD No. _____
DDC FILE COPY

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited



ARMY FORGE OFFICE OF SCIENTIFIC RESEARCH (AFSOF)
NOTICE OF TRANSMITTAL TO DPM
This technical report has been reviewed and is
approved for publication in accordance with AFM 190-12 (7b).
Distribution is unlimited.
A. D. BLOSE
Technical Information Officer

2

TR-648
AFOSR-77-3271

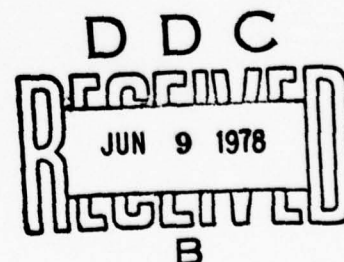
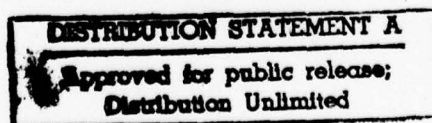
April 1978

CELLULAR GRAPH ACCEPTORS, 3

Angela Wu
Computer Science Center
University of Maryland
College Park, MD 20742

ABSTRACT

In earlier reports, cellular acceptors were studied whose languages are sets of d -graphs, i.e., labelled graphs of bounded degree whose arcs at each node are numbered. This report discusses acceptance tasks that depend on the concept of d -graph isomorphism -- in particular, the task of deciding whether a d -graph has a d -subgraph isomorphic to a given d -graph.



The support of the Directorate of Mathematical and Information Sciences, U. S. Air Force Office of Scientific Research, under Grant AFOSR-77-3271, is gratefully acknowledge, as is the help of Mrs. Shelly Rowe in preparing this paper.

1. Introduction

Cellular acceptors whose languages are sets of d-graphs (labelled graphs of bounded degree whose arcs at each node are numbered) were studied in [1-2], where the terminology and notation used in the present paper are defined. This paper discusses acceptance tasks that depend on the concept of d-graph isomorphism -- in particular, the task of deciding whether a d-graph has a d-subgraph isomorphic to a given d-graph.

Given two node labelled graphs $\gamma_1 = (N_1, A_1, f_1)$ and $\gamma_2 = (N_2, A_2, f_2)$ where f_1, f_2 are the node labelling functions, γ_1 is isomorphic to γ_2 if there exists a bijection b from N_1 to N_2 such that $f_1(n) = f_2(b(n)) \forall n \in N_1$ and $(m, n) \in A_1$ iff. $(b(m), b(n)) \in A_2$. A d_1 -graph $\Gamma_1 = (N_1, A_1, f_1, g_1)$ and a d_2 -graph $\Gamma_2 = (N_2, A_2, f_2, g_2)$ are isomorphic (denoted by $\Gamma_1 \approx \Gamma_2$) iff. their underlying graphs $U(\Gamma_1)$ and $U(\Gamma_2)$ are isomorphic. Here we allow $d_1 \neq d_2$. A subgraph of a d-graph $\Gamma = (N, A, f, g)$ is denoted by $(N', A', f|N', g|A')$ where $N' \subseteq N$ and $A' \subseteq A$ and if $(m, n) \in A'$ then $m \in N'$ and $n \in N'$. Note that $(N', A', f|N', g|A')$ is not necessarily a d-graph, since some of the nodes may not have exactly d neighbors. However, we can always attach # nodes so as to make it into a d-graph. A labelled graph α is isomorphic to Γ if $\alpha \approx U(\Gamma)$, and α is isomorphic to a subgraph of Γ if $\alpha \approx U(\Gamma')$ for some subgraph Γ' of Γ . In the following, we will consider only connected d-graphs.

BY		
EXEMPTION/AVAILABILITY CODE		
Dist.	AVAIL.	and/or SPECIAL
A		

2. Graph isomorphism

In this section, we will consider acceptance tasks that depend on graph isomorphism. Specifically, given a labelled graph α of degree $\leq d$, we will find a finite state acceptor M_α such that (Γ, M_α, H) accepts Γ iff. α is isomorphic to Γ and rejects Γ otherwise.

We first need

Proposition 1. For every integer $r > 0$, there is a finite state acceptor M_r such that the cellular d -graph acceptor (Γ, M_r, H) with distinguished node D accepts all d -graphs Γ whose nodes are all within distance r from D in $2r+1$ steps, and when it accepts, every node is in a different state.

Proof: Given any d -graph Γ , the cellular d -graph acceptor (Γ, M_r, H) operates as follows: the distinguished node D sends out a message S which propagates to the nodes at distance r from D . The paths traveled by S define a spanning tree of Γ , and each node is identified uniquely by marking each node's state with a sequence of arc end numbers which define the unique path from D to the node. Specifically, when a neighbor of D receives S , its state is marked with the number i if it is the i th neighbor of D . It then sends the message (S, i) to its neighbors. When an unmarked node m receives the message (S, i_1, \dots, i_k) , $k \geq 1$, from node n , and m is the j th neighbor of n , then m marks its state with (i_1, \dots, i_k, j) and sends (S, i_1, \dots, i_k, j) to its neighbors. If a node receives a message from more than one neighbor

simultaneously, it can choose to accept one of them, say the one sent by the lowest-numbered neighbor. Since the paths from D to each node are all different, the sequences of numbers in the states of the nodes are distinct.

If a node m_1 is marked with (i_1, i_2, \dots, i_r) and one of its neighbors, say m_2 , is still unmarked, then m_2 is at distance $r+1$ away from D . A rejection signal is thus sent to D because the graph contains nodes more than distance r away. If no rejection signal is received after $2r+1$ steps, Γ is accepted. //

Given a node-labelled graph α of degree $\leq d$, we can find its diameter r and construct its spanning tree T_α using the method in Section 1.3.1 of [2]. The height of the spanning tree is $\leq r$ and associated with each node is a level number. The level numbers of a node and its neighbors differ by at most 1; this follows from the way the tree is constructed. Now we can prove

Proposition 2. For any labelled graph α of degree $\leq d$, there exists an M_α such that the cellular d -graph acceptor (Γ, M, H) with distinguished node D accepts Γ if $\alpha \cong \Gamma$ and rejects Γ otherwise.

Proof: M_α first simulates the action of M_r , where r is the diameter of α , in the first r steps. It sends a rejection signal to the distinguished node D if it finds a node at distance more than r from D , since in this case Γ cannot be isomorphic to α . At the end of step r , every node of Γ has

a unique identity represented in its state.

Step $r+1+i$ ($0 \leq i \leq h$) identifies the nodes corresponding to level $k-i$ nodes of T_α based only on the knowledge that the node has the right neighbors to serve as its sons in T_α . In the states of these nodes, the numbers of the nodes that are to be its descendants are recorded. Thus at the end of step $r+1+h$, the nodes indicating that they can be the root of T_α are saying that the assignments in their states are sure that all the arcs of T_α exist, but the arcs in α and not in T_α will have to be checked. More specifically,

Step $r+1$: Each node decides if it can be a level h node of T_α by looking at its label. A node that is qualified indicates this fact by recording in its state $([n_1, id], \dots, [n_k, id])$, $n_i \neq n_j$ if $i \neq j$, where id is the unique identity of the node obtained in previous steps, and the n_i 's are the possible level h nodes it can be. The number k is at most the number of level h nodes of T_α , so the length of the state is bounded. All other nodes are in some neutral state.

Step $r+2$: Each node looks at its neighbors. If it has the right level h neighbors to serve as its sons in T_α and qualify it to be a level $h-1$ node, then it changes its state to $([n_1, id], (n_{11}, id_{11}), \dots, (n_{1i_1}, id_{1i_1})),$

$$[(n_2, id), (n_{21}, id_{21}), \dots, (n_{2i_2}, id_{2i_2})],$$

$$\vdots$$

$$[(n_k, id), (n_{k1}, id_{k1}), \dots, (n_{ki_k}, id_{ki_k})]$$

Here $id \neq id_{ij} \forall i, j$ and $id_{ij} \neq id_{i'j'}$ if $j \neq j'$;

id is the unique identity of the node. Each

$[(n_j, id), \dots, (n_{ji_j}, id_{ji_j})]$ indicates that

this node can be node n_j of T_α , the subtree

of T_α at n_j consists of nodes n_{j1}, \dots, n_{ji_j} ,

and they correspond to nodes with identities

$id_{j1}, \dots, id_{ji_j}$, respectively. For each node,

the numbers of level $h-1$ nodes it can be

is bounded and the possible assignment from

its neighbors to the subtree of T_α is also

bounded. Therefore the length of the states

is bounded. All other nodes are in some

neutral state regardless of their previous

states.

$$\vdots$$

Step $r+1+i$: The qualified level $h-i$ nodes record in

their states the assignment of nodes to serve

as the subtree of T_α at the level $h-i$ node

it qualifies to be. Of course an assignment

is made only if no node in the assignment

corresponds to two different nodes of α .

$$\vdots$$

Step $r+1+h$: The possible root nodes with the assignments of all the nodes of T_α are known. These assignments are based on the knowledge that the arcs in T_α exist. Therefore each of these assignments gives a subgraph of Γ isomorphic to T_α .

Starting at Step $r+h+2$, each qualified root node initiates signals to check each assignment recorded in its state to make sure that all the arcs in α exist and no other arcs are present. This is done by transmitting the assignment to each node. If a node not in the assignment receives the assignment signal, this means that Γ has more nodes than α and it cannot be isomorphic to α ; thus a rejection signal is sent to the distinguished node to reject Γ . If a node is in the assignment, when it receives the signal, it makes sure that all the arcs incident upon it are connected to the nodes with the correct identities as in α . Any time a node finds an arc out of order, it sends a cancellation signal to report to the root node of this assignment to delete the assignment. If after $2h$ steps, the qualified root node finds that it still has uncanceled assignments, then it can send a success signal to the distinguished node D . When D gets a success signal, it accepts. However, if at the end of step $r+1+h+2h+r$ no success signal is received by D , this means there is no successful assignment. This is because either the assignments made at step $r+1+h$ are all cancelled or there

is no assignment at all at step $r+1+h$, since there may be too few nodes in Γ , or it is not possible even to find a subgraph of Γ isomorphic to T_α . In any case, Γ is not isomorphic to α and Γ is rejected.

An alternate method to test for isomorphism is: first give each node a unique identity as above, but starting at step $r+1$, the nodes' identities are transmitted to and collected by the distinguished node D . At the same time, the number of non-# nodes in Γ is counted and compared with k , the number of nodes in α . If these two numbers are not equal, D rejects Γ ; otherwise D sends out $k!$ signals, each specifying an assignment of the node identities of the nodes to the nodes of α . These signals propagate from neighbor to neighbor. As in the other method, each node checks its arcs as the signals are received and sends cancellation messages if any incident arc is not as in α . $2r$ steps after D sent out the $k!$ messages, if there is still an assignment at D , then it accepts Γ , otherwise it rejects. Since counting and propagation of the messages all take order diameter time, this method also detects graph isomorphism in diameter time.//

3. Subgraph isomorphism

3.1 k-level-colored d-graphs

In this section we consider the diameter time subgraph matching problem: Given a labelled graph α , find an M_α such that (Γ, M_α, H) accepts Γ iff. α is isomorphic to a subgraph of Γ in time proportional to the diameter of Γ . Unlike the graph isomorphism case, the number of nodes of the d-graph Γ may be arbitrarily large. However, the definition of M_α depends only on α and not on Γ ; therefore it is not possible to give each node of Γ a unique identification as part of its state.

Suppose α has diameter r ; if a node n is part of a subgraph S isomorphic to α , then all the other nodes of S must be within distance r from n . We will show that if every node of Γ has a different state from any node within distance r from it, then we can discover whether α is isomorphic to a subgraph of Γ in time proportional to the diameter of Γ .

A d-graph Γ will be called k-level-colored if the nodes of Γ are colored and any two nodes within distance k from each other have different colors. For any node n in a d-graph Γ , the number of nodes within distance k from it is at most $c(k) = d + d(d-1) + d(d-1)^2 + \dots + d(d-1)^{k-1}$. The number of colors needed for Γ to be k-level-colored is no more than $1+c(k)$. We can assume that the color at each node is part of the label and thus becomes part of the initial state of the automaton at the node, and that the color of a node is its identity.

Proposition 3. Given a labelled graph β with diameter r , there is a finite state automaton M_β such that for any k -level-colored ($k \geq r$) d -graph Γ , the cellular d -graph acceptor (Γ, M_β, H) with a distinguished node D accepts Γ if $\beta \approx a$ subgraph of Γ , and rejects Γ otherwise, in time proportional to the diameter of Γ .

Proof: Given β we can find its spanning tree T_β . Let h be the height of T_β . M_β works in almost the same way as M_α in Section 1 except that the first r steps of M_α are not necessary since each node already has an identity. In the process of identifying nodes of various levels, no assignments can be made that requires two nodes of β to correspond to the same identity, because nodes in the same assignment must be within distance r from each other so that only one node can have a particular identity. After $h+1$ steps, all the possible root nodes with the correspondence between node of T_β and the node identities are known. M_β again initiates signals to check for the existence of the arcs in β but not in T_β as M_α did with some slight modifications. The signal corresponding to each assignment is sent and transmitted from neighbor to neighbor, but it stops propagating after r steps so that it will not reach nodes at distance more than r away. When a node in the assignment receives the signal, it makes sure that the corresponding arcs in β incident to it all exist and are connected to the nodes with the correct identities. It does nothing if those arcs exist, but if one of the arcs

is out of order, it sends a cancellation signal to the supposed root node to delete that assignment. Again, this signal travels only h steps. All the assignments not cancelled after $2h$ steps are good. Thus at that point, if a supposed root node finds that it still has an uncanceled assignment, it sends a success signal to the distinguished node D . When D gets a success signal, it accepts. However, if at the end of step $p = h + 1 + 2h + (\text{radius of } \Gamma \text{ centered at } D)$ and no success signal is received by D , it means there is no successful assignment and no subgraph of Γ is isomorphic to β . Therefore we can reject Γ . Note that no success signal reaches D at step p iff. no success signal reaches D after step p . Therefore at step 1, besides working in the same way as M_α , M_β also sends out a special signal F from D . A return signal R is sent back to D when F reaches the leaf nodes as in the spanning tree construction of Section 1.3.1 in [2]. It takes twice the radius of Γ for D to receive R from all of its neighbors. Then D waits for another $3h + 1$ steps; if no success signal is received, it rejects. //

It should be pointed out here that each uncanceled assignment represents a subgraph isomorphic to β . Moreover there are redundancies because the same subgraph may be specified by different assignments which correspond to different automorphic images of the subgraph.

3.2 Trees

Proposition 4. For any $k > 0$, there is a finite state automaton M_k such that for any d-graph Γ which is also a tree, in time proportional to the diameter of Γ , the cellular d-graph automaton (Γ, M_k, H) k-level-colors the nodes of Γ , i.e., the states of nodes within distance k from each other are distinct.

Proof: Let $\ell = \lfloor \frac{k}{2} \rfloor$ whose $|a|$ = the largest integer $\leq a$. Each node's state will have a component of the form (i, i_1, \dots, i_ℓ) where $0 \leq i \leq k$, $0 \leq i_j \leq d \ \forall 1 \leq j \leq \ell$, which serves as the color of the node. At the first step the distinguished node writes $(0, 0, 0, \dots, 0)$ in its state and sends this message to its neighbors. When an uncolored node n receives a message (i, i_1, \dots, i_ℓ) from its neighbor m , node n writes in its state $(i+1 \text{ (modulo } k+1), i_2, i_3, \dots, i_\ell, j)$ if m is the j th neighbor of n , and n sends this message to its neighbors. If an uncolored node receives messages from more than one node simultaneously, it chooses only one of them. When a node n is colored with (j, j_1, \dots, j_ℓ) , this says that if D is the root, n is a node at level $(k+1)i+j$ for some $i \geq 0$ and j_1, \dots, j_ℓ specifies a path from its ancestor at level $(k+1)i+j-\ell$ if $j_1 \neq 0$. If j_1 is 0, then $j_h, j_{h+1}, \dots, j_\ell$ is a path from D where j_h is the first nonzero element after j_1 .

Given any two nodes n_i and n_j within distance k from each other, suppose their colors are $c(n_i) = (h_i, i_1, \dots, i_\ell)$ and $c(n_j) = (h_j, j_1, \dots, j_\ell)$ respectively. If $h_i \neq h_j$, then $c(n_i) \neq c(n_j)$. If $h_i = h_j$ then n_i and n_j are on the same

level of Γ with root D because the levels of n_i and n_j are $(k+1)t_1 + h_i$ and $(k+1)t_2 + h_j$ for some t_1, t_2 ; their difference is $\leq k$ (otherwise n_i is not within distance k from n_j), and this implies $t_1 = t_2$. Let n be the closest common ancestor of n_i and n_j , i.e., no descendent of n is an ancestor of both n_i and n_j . n is at equal distance from n_i and n_j . The distance between n and n_i or n_j is $\leq \ell = \lfloor \frac{k}{2} \rfloor$, since otherwise the distance between n_i and n_j is $> k$. Therefore $(i_1, \dots, i_\ell) \neq (j_1, \dots, j_\ell)$ since one contains a path from n to n_i and the other a path from n to n_j . The time it takes for the signal from D to reach a node n equals the distance between D and n . Therefore the coloring process takes diameter (Γ) time. //

Combining the results of Propositions 3 and 4, we see that if a d -graph Γ is a tree, then we can do subgraph matching in diameter (Γ) time. However, because of the structure of trees -- they have no cycles, and between any two nodes there is only one simple path we can also do subgraph matching without first coloring the nodes.

Let Γ be a d -graph which is also a tree. Then any labelled graph isomorphic to a subgraph of Γ must be a tree. Let Z be a tree with root node r and height h . In the remainder of this section, we describe a cellular d -graph automaton (Γ, M_Z, H) which accepts Γ iff. $Z \approx$ a subgraph of Γ . The action of M_Z is similar to the actions of M_α and M_β in the previous sections. Namely, at each step i ($1 \leq i \leq h+1$)

each node n looks at its neighbors to decide if they can correspond to the sons of a level $h+1-i$ node m in the tree Z . If they can, n declares itself to qualify as the node m of Z . The difference is that the nodes do not have distinct identities any more; rather, we know that Γ is a tree. Instead of recording the assignments of nodes, each node n 's state simply indicates the level $h-i+1$ node it can be and the arc end numbers leading to its sons. Note that a node can correspond to a few different level $h-i+1$ nodes with different sets of sons. In the next step, the knowledge of the sons prevents those nodes from serving both as n 's father and son when n corresponds to a particular node m_0 of Z since n 's neighbor can see from n 's state whether it was used as n 's son to qualify n as m_0 of Z .

Claim: At the end of step i ($1 \leq i \leq h+1$), if a node n 's state indicating that it qualifies as a level $h-i+1$ node m of Z and its i_1, i_2, \dots, i_j -th neighbors n_1, n_2, \dots, n_j correspond to the sons m_1, m_2, \dots, m_j of m in Z , then n is the root of a tree* isomorphic to the subtree of Z at m .

Proof: When $i=1$, the claim is clearly true since the subtree of Z at a leaf node consists of only the node itself. If the claim is true for $i-1$ ($1 \leq i \leq h+1$), then n_1, n_2, \dots, n_j are all roots of trees isomorphic to subtrees of Z at m_1, \dots, m_j . First note that if n qualifies as corresponding

*This tree is an acyclic subgraph of Γ .

to m , then n qualifies to be the father of n_1, \dots, n_j . In the trees isomorphic to subtrees of m_1, \dots, m_j , all of the n_k 's ($1 \leq k \leq j$) do not use n as a son, since otherwise n could not be that n_k 's father. Moreover, the trees at n_1, n_2, \dots, n_j are all disjoint since they are all subgraphs of Γ and Γ is a tree. If a node A belongs to both the trees at n_1 and n_2 , then there exist a sequence of nodes $n_1, a_1, \dots, a_\ell, A$ joining n_1 to A , and a sequence of nodes n_2, b_1, \dots, b_k, A joining n_2 to A . This means that $n, n_1, a_1, \dots, a_\ell, A, b_k, \dots, b_1, n_2, n$ is a cycle, a contradiction. This shows that n is the root of a tree. This tree is isomorphic to the subtree of Z at m ; this is clear from the fact that the trees at n_k are isomorphic to subtrees at m_k ($1 \leq k \leq j$). //

At the end of step $h+1$, all the nodes that correspond to the root node of Z are identified. These nodes just send a success message to the distinguished node D for acceptance. If after $h+1+\text{height}(\Gamma)$ steps, no success message is received by D , it rejects. Again M_Z can use the same method as M_β of Proposition 3 to decide when to reject.

We have thus proved

Proposition 5. For any labelled tree Z of height h , there is a finite state automaton M_Z such that for any d -graph Γ which is a tree, the cellular d -graph automaton (Γ, M_Z, H) accepts Γ iff. $T \approx$ a subgraph of Γ , and otherwise it rejects Γ , in time proportional to the diameter of Γ .

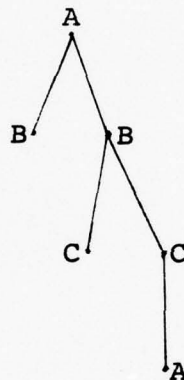
In Section 1.2.3 of [2], we showed how a cellular d -

graph automaton recognizes trees in diameter time. Combining this with the above proposition, we have the result that for any tree Z , there is a cellular d-graph automaton that recognizes all the d-graphs which are trees and have subgraphs isomorphic to Z .

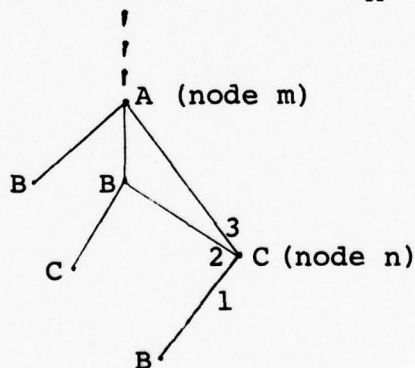
3.3 k-local-homogeneous d-graphs

In the last section, we exhibited a diameter time subgraph matching algorithm for trees without appealing to k-level-coloring. We have been unable to find diameter time algorithms for general d-graphs. The problems seem to be due to the existence of cycles in a general d-graph, as can be seen from the following examples:

Example 1: Suppose the labelled graph α is



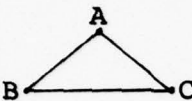
and part of Γ is:

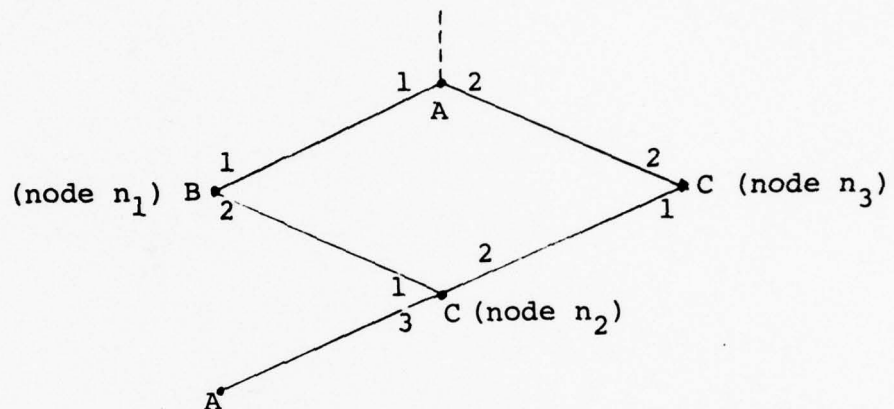


If the states of the nodes are not different, there seems to be no way that node n can tell that its third neighbor cannot serve as its son since it is already used as its ancestor. Equivalently, there seems to be no way to prevent node m from

serving as two different nodes of α .

Example 2: Suppose we are looking for subgraphs to match


 . Suppose that part of Γ is



How can node n_1 tell that node n_2 is not node n_3 (and thus not the desired structure) if the states of the nodes are not different? In general, how can a node distinguish the signals from different nodes? This is the same problem as in [3, 4] where the automaton cannot tell apart cycles of lengths 3 and 4.

Let us examine the special properties a tree has:

- (1) There are no nontrivial cycles in a tree.

The only cycles are those consisting of a path and its exact inverse.

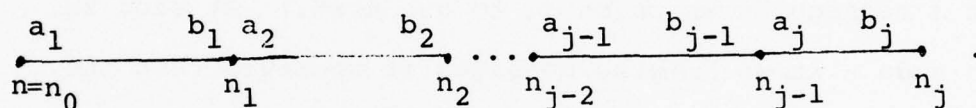
- (2) There is only one path between any two points.

In this section, we discuss a generalization of

these properties, and define a class of d-graphs for which diameter time subgraph matching is possible.

3.3.1 Definition of k-local-homogeneity

At each node n , $H(n) = (t_1, \dots, t_d)$ tells n that it is the t_i th neighbor of its i th neighbor. If we consider a sequence of numbers a_1, \dots, a_j ($1 \leq a_i \leq d$ for $1 \leq i \leq j$) at n as a path $n = n_0, n_1, \dots, n_j$ such that n_i is the a_i th neighbor of n_{i-1} ($1 \leq i \leq j$)*, then $H(n)$ tells node n the inverse of any path of length 1. Define $H^j(n): D^j \rightarrow D^j$ such that if the image of (a_1, \dots, a_j) is (b_1, \dots, b_j) then the inverse of the path a_1, \dots, a_j is b_j, b_{j-1}, \dots, b_1 and we have



It is obvious that knowing H^k at a node n implies knowing H^j at n for any $1 \leq j \leq k$.

Proposition 6. For any $k > 0$, there exists a finite state automaton M_k such that for any d -graph Γ , the cellular d -graph automaton (Γ, M, H) can find H^k and record it in its state in $2k$ steps.

Proof: Since each node knows its H -function, it can send out messages in the form of $(i; t_i)$ to its i th neighbor ($1 \leq i \leq d$). When a node m receives $(i; t_i)$ from its t_i th neighbor, it

*From now on, we will use the notations " $n = n_0, \dots, n_j$ " and " a_1, \dots, a_j at n " interchangeably.

sends out messages $(i, j; t_1, t_j')$ to its j th neighbor n_j ($1 \leq j \leq d$) where n is the t_j' neighbor of n_j . In general, for $\ell < k$, when a node n receives $(i_1, i_2, \dots, i_\ell; t_1, t_2, \dots, t_\ell)$ from its t_ℓ th neighbor, it augments the message and sends out $(i_1, i_2, \dots, i_\ell, j; t_1, \dots, t_\ell, t_j')$ to its j th neighbor ($1 \leq j \leq d$) where n is the t_j' th neighbor of its j th neighbor. If n 's j th neighbor is a # node, it makes a special mark. After step k , instead of augmenting and sending out new messages, the messages backtrack, i.e., the message $(i_1, i_2, \dots, i_k; j_1, j_2, \dots, j_k)$ travels along the path j_k, \dots, j_2, j_1 . (A pointer is kept so that a message knows which j_ℓ to use next.) At step $2k$, each node n knows from the messages it receives that the image of (i_1, \dots, i_k) at n under H^k is (j_1, \dots, j_k) . Hence the inverse of the path i_1, \dots, i_k from n is j_k, \dots, j_1 . If a path does not reach length k because a # node is encountered, the node can tell this from the special mark at that position. Note that since each message only travels distance k away and each node has at most $d + d(d-1) + \dots + d(d-1)^{k-1}$ nodes within distance k from it, the number of signals at any node at each step is bounded. //

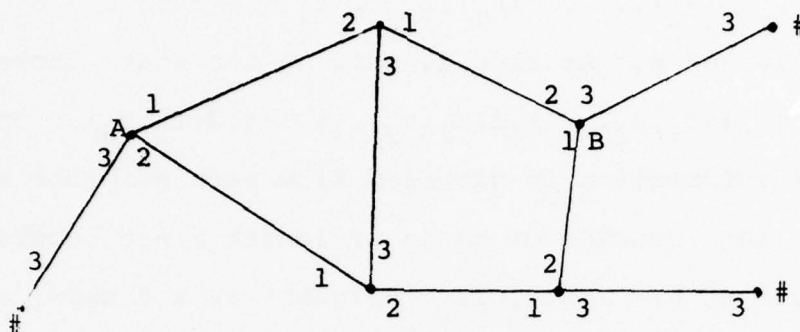
Alternate proof: At the first step, each node writes in its state the inverses of paths of length 1 from it, i.e., records its H -function. At the next step, since each node can see its neighbor's states, it can tell the inverses of paths of length 2. More specifically, suppose $H(n) = (t_1, t_2, \dots, t_d)$, $H(m) = (s_1, s_2, \dots, s_d)$ and m is the

ith neighbor of n . At the end of step 1, n 's state has $((1;t_1), (2;t_2), \dots, (d;t_d))$ and m 's state has $((1;s_1), (2;s_2), \dots, (d;s_d))$. If $t_i = j$ then $i = s_j$ by the definition of H . At step 2, part of n 's state looks like this: $(i,1;t_i,s_1), (i,2;t_i,s_2), \dots, (i,d;t_i,s_d)$. The same kind of information is gathered from each neighbor of n , so that the inverses of paths of length 2 are recorded in the state of n . Again, if a neighbor is a # node, a special mark is recorded in the state so that it will not attempt to extend that path further. Therefore, at each step, we can use the latest information each node acquired at the previous step to extend the paths' lengths by 1. At the end of step k , each node's state has H^k coded in it. //

In a d -graph Γ , a node n knows all cycles of length up to k if given a sequence of arc end numbers a_1, \dots, a_j ($1 \leq a_j \leq d$, $1 \leq j \leq k$), hence a path of length $j \leq k$ starting from n , node n knows whether or not this path is a cycle, i.e., $n = n_0, n_1, \dots, n_{j-1}, n_j = n$, where n_i is the a_i the neighbor of n_{i-1} ($1 \leq i \leq j$). A node n knows all equivalent paths of total length up to k if given any two sequences of arc end numbers, hence two paths from n , with sum of their lengths $\leq k$, node n knows if they lead to the same node. Γ is said to know all cycles of length up to k or know all equivalent paths of total length up to k iff. every node in Γ knows the respective information. The following example shows that knowing H^k does not imply knowing all cycles of length j nor knowing

all equivalent paths of total length j for some $j > 2$.

Example:



$H(n) = (2, 1, 3)$ for every node n and so $H^k(n)$ is the same for every node. However, the cycles at node A and node B are quite different. At A, 232 gives a cycle, while at B, 232 is not a cycle. At A, 23 and 1 reach the same node, but at B, 23 and 1 do not meet.

Proposition 7. A node n knows all cycles of length up to k iff. n knows all equivalent paths of total length up to k .

Proof: For any d -graph Γ , there is a cellular d -graph automaton that finds H^k in constant time (depending only on k). Suppose n knows all cycles of length up to k . Given any two paths P_1, P_2 whose total length $\leq k$, n knows H^k ; thus it can invert one of the paths, say the shorter one P_2 , and append the inverse of P_2 to P_1 . The result is a sequence of arc end numbers of length $\leq k$. This sequence is a cycle iff. P_1 and P_2 reach the same node. But n knows all cycles of length up to k , therefore n knows all equivalent paths of total length up to k .

Conversely, suppose n knows all equivalent paths of total length up to k . Given a sequence of arc end numbers (a_i) of length $\leq k$, n can simply break it at some point into two, using its knowledge of H^k to find the inverse of one of them. This yields two paths at n , the sum of their lengths = length of $(a_i) \leq k$. (a_i) is a cycle iff. these two paths reach the same node. But n knows all the equivalent paths of total length up to k ; therefore n knows all the cycles of length up to k .//

Having the equivalence given by Proposition 7, we can define a d-graph Γ to be k-locally-homogeneous if at each node n of Γ , $H(n)$ and all cycles of length up to k or all equivalent paths of total length up to k are known. Clearly, a d-graph that is a tree is k-locally-homogeneous, since there are no cycles, and no two distinct paths can be equivalent.

3.3.2 Subgraph matching problem for k-locally-homogeneous d-graphs

Let ω be a labelled graph such that the degrees of the nodes of ω are $\leq d$ and the diameter of ω is $r > 0^*$. We will define a deterministic cellular d-graph acceptor with finite-state automaton M_ω that will recognize those k-locally-homogeneous d-graphs ($k > 2r$) having a subgraph isomorphic to ω . For each ω , taking any one of the nodes as the root node and using the methods described in [2], a spanning tree T_ω of ω can be constructed, where the height of T_ω is $h \leq r$.

The diameter time matching of ω to a subgraph of any k-locally-homogeneous d-graph Γ will be done by first trying to identify the subgraphs of Γ isomorphic to T_ω . These subgraphs will be stored in the state of the node corresponding to the root of T_ω in the form of specifying arc end numbers leading to each node corresponding to a node of T_ω . Then using homogeneity conditions, the nonspanning tree edges of ω can be checked.

In identifying subgraphs of $\Gamma \approx T_\omega$, the action of M_ω is similar to the actions of M_α , M_β and M_Z defined earlier. Namely, at each step i ($1 \leq i \leq h+1$), each node decides if it can correspond to a node m at level $h+1-i$ of T_ω by checking if its neighbors can correspond to the sons

*If $r=0$ then ω has only one node; subgraph matching then becomes the label detecting problem of [2].

of m in T . However, the nodes are no longer colored distinctly and Γ is not necessarily a tree. Therefore, each node n records in its state the nodes of T_ω that it qualifies to be and the arc end numbers leading to the neighbors corresponding to the sons of n , together with the information recorded in each of those neighbors. More specifically,

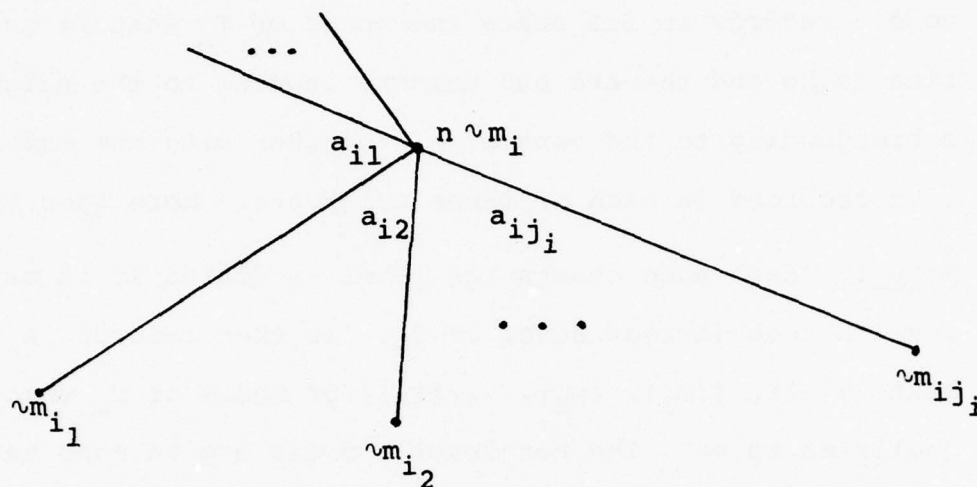
Step 1: Each node checks its label to decide if it can be a level h node (a leaf node) of T_ω . It then records in its state a list $([m_1], [m_2], \dots, [m_j])$ of nodes of T_ω that it qualifies to be. The non level h nodes are in some neutral state.

Step 2: Each node n with the proper label, and having neighbors which can correspond to the sons of a level $h-1$ node, writes in its state:

$$\begin{aligned}
 &([m_1, (a_{11}, [m_{11}]), (a_{12}, [m_{12}]), \dots, (a_{1j_1}, [m_{1j_1}])], \\
 &[m_2, (a_{21}, [m_{21}]), \dots, (a_{2j_2}, [m_{2j_2}])], \dots \\
 &[m_s, (a_{s1}, [m_{s1}]), \dots, (a_{sj_s}, [m_{sj_s}])])
 \end{aligned}$$

where $a_{it} \neq a_{it'}$, if $t \neq t'$. Here m_1, m_2, \dots, m_s are the level $h-1$ nodes of T_ω that n can correspond to, and m_{ij} are the level h nodes of T_ω . $[m_i, (a_{i1}, [m_{i1}]), \dots, (a_{ij_i}, [m_{ij_i}])]$ means that if n corresponds to node m_i , then its a_{it} -th neighbor corresponds to node m_{it} of T_ω where m_{it} ($1 \leq t \leq j_i$) are the sons of m_i in T_ω . This specifies a subgraph of Γ as shown below. Obviously, the m_i 's are not necessarily distinct. All the other non level $h-1$ nodes are changed to the neutral

state regardless of what state they were in.

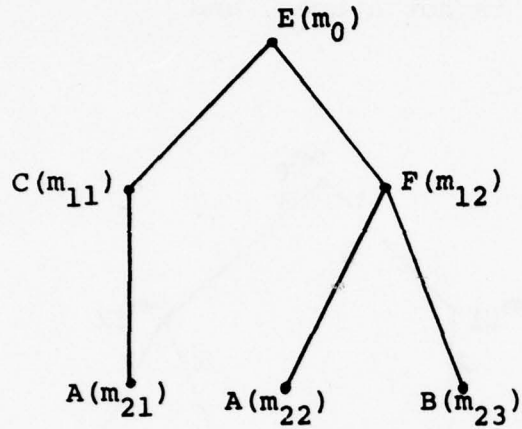


In general, at step i , $1 \leq i \leq h+1$, a node n that qualifies to be a level $h+1-i$ level node changes to a state of the form (S_1, S_2, \dots, S_p) where each S_j specifies a subgraph of Γ at n and $S_j = [m_j, (b_{j1}, A_{j1}), \dots, (b_{ji_j}, A_{ji_j})]$ where A_{jt} is $[m_{jt}]$ or $[m', (a_{p1}, A'_{p1}) \dots (a_{ps}, A'_{ps})]$ and A_{jt} is in the state of the b_{jt} -th neighbor of n .

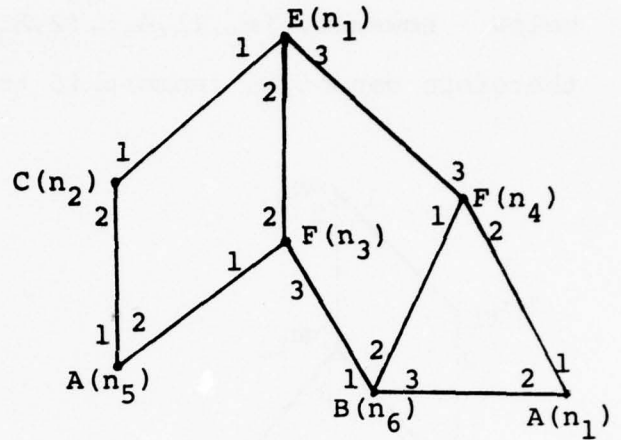
At step $h+1$, since T_ω has only one node at level 0, all the nodes that can possibly correspond to the root of a subgraph isomorphic to T_ω have that subgraph recorded in their states. However, as shown by the following example, this subgraph is not necessarily a tree.

Example: Let T_ω and Γ be the graphs shown below.

T_ω :



Γ :



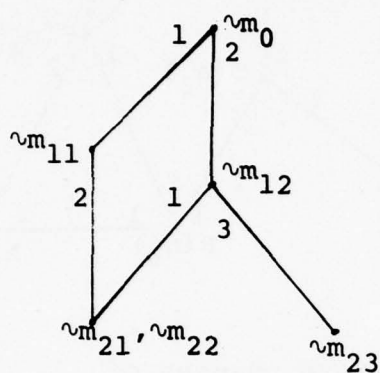
The following table summarizes the state changes of the nodes of Γ at each step:

Node	Step 1	Step 2	Step 3
n_1	b	b	$([m_0, (1, A_2), (2, A_3)], [m_0, (1, A_2), (3, A_4)])$
n_2	b	$([m_{11}, (2, [m_{21}])])$	b
n_3	b	$([m_{12}, (1, [m_{22}]), (3, [m_{23}])])$	b
n_4	b	$([m_{12}, (1, [m_{23}]), (2, [m_{22}])])$	b
n_5	$([m_{21}], [m_{22}])$	b	b
n_6	$([m_{23}])$	b	b
n_7	$([m_{21}], [m_{23}])$	b	b

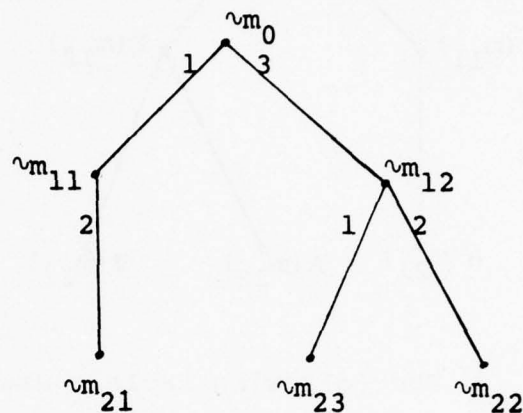
where A_2, A_3, A_4 are the states of n_2, n_3, n_4 at step 2.

The state of n_1 shows that there are two subgraphs as shown

below. However, $[m_0, (1, A_2), (2, A_3)]$ is not a tree, and therefore cannot be isomorphic to T_ω .



$[m_0, (1, A_2), (2, A_3)]$



$[m_0, (1, A_2), (3, A_4)]$

It is easy to see (by an induction proof) that if the subgraph S obtained at a node n at step $h+1$ is a tree, then S is isomorphic to T_ω . S is a tree iff. it contains no cycles. Hence, since Γ is k -locally-homogeneous, n can tell which subgraph in its state is not a tree, and can delete that subgraph from its state at step $h+2$. All the subgraphs that remain are trees isomorphic to T_ω . [In fact, the task of cycle checking can be done so that each node declares itself to be a level $h+1-i$ node only if its subgraph is a tree, so that many non-tree subgraphs are deleted before step $h+1$.] At step $h+3$, node n checks to make sure the edges of ω not in T_ω exist, using the knowledge it possesses about k -local homogeneity. For example, suppose m_1 should

be connected to m_j in ω , and the tree path in the subgraph from the root node n to the corresponding nodes of m_i and m_j are a_1, \dots, a_{t_i} and b_1, \dots, b_{t_j} ; then node n just has to make sure that there is an arc end number c such that a_1, \dots, a_{t_i}, c and b_1, \dots, b_{t_j} both reach m_j . If any non-tree edge does not exist, that subgraph is deleted from the state of node n .

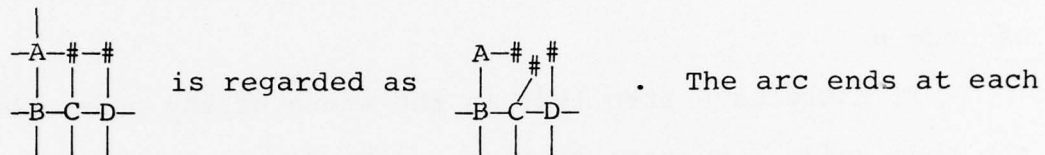
If there is a tree left in the state of any node, then the node sends a success message to the distinguished node to signal acceptance. If the distinguished node receives no success message after step $h+3+\text{diameter}(\Gamma)$ it rejects Γ . Again the distinguished node can tell that $h+3+\text{diameter}(\Gamma)$ steps have passed by the same method used in Proposition 3.

We have thus proved

Proposition 8. For any labelled graph ω with degree $\leq d$ and diameter r , there exists a finite state automaton M such that for any k -locally homogeneous d -graph Γ ($k > 2r$), the cellular d -graph acceptor (Γ, M_ω, H) accepts Γ if $\omega \approx$ a subgraph of Γ , and rejects Γ otherwise, in time proportional to the diameter of Γ .

3.4 Homogeneous d-graphs

A two-dimensional array may be regarded as a 4-graph, provided we assume the boundary (#) nodes are distinct so that each # node has only one neighbor, i.e.,



node are labelled with 1(=N), 2(=W), 3(=S), 4(=E). Each node n knows the inverse of any path starting from n , since 1 and 3, 2 and 4 are always inverses of one another. Each node also knows when a path is a cycle by checking if the number of 1's = the number of 3's and the number of 2's = the number of 4's. Therefore a two-dimensional array is a k -locally-homogeneous d-graph for any $k \geq 1$. Moreover, all the k -local-homogeneity conditions at each node are the same in the sense that if a path from a node exists (no # node is encountered) then the same criterion determines, for all nodes, whether or not the path is a cycle.

A d-graph will be called k -homogeneous if all the k -local-homogeneity conditions are the same for every node of Γ . If the d-graph is k -homogeneous for every $k \geq 1$, we call it simply homogeneous. As indicated above, the two-dimensional arrays are homogeneous 4-graphs. It is easy to see, analogously, that any n -dimensional array is a homogeneous $2n$ -graph.

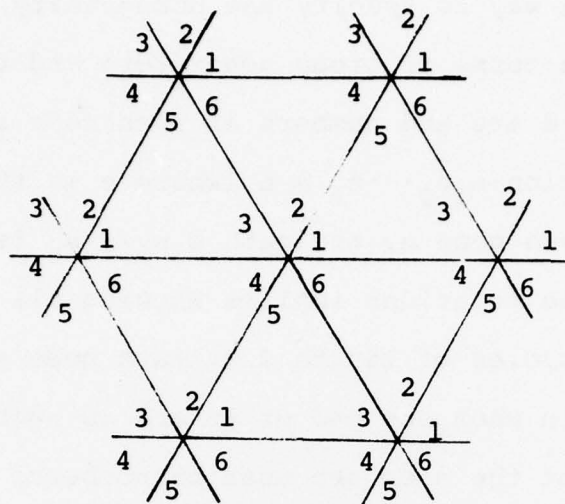
A natural way to specify the homogeneity conditions of a d -graph is in terms of group generators and relations. We can regard the d arc end numbers at each node as the generators. A relation $s_1 s_2 \cdots s_t = e$ (where e is the identity) says that at each node n , the path $s_1 s_2 \cdots s_t$ is a cycle. Thus knowing the relations implies knowing all the cycles. Moreover, the cycles of length 2 at each node are the same. If $s_1 s_2 = e$ then when one end of an arc is numbered with s_1 , the other end of the same arc must be numbered with s_2 ; thus $s_1 = s_2^{-1}$. This shows that the d generators must form a group.

Mylopoulos and Pavlidis in [5, 6] described a number of graphs corresponding to different finitely presented abelian groups. Any finite subgraph of one of these graphs (with the appropriate # nodes added to make the degree exactly d at each non-# node) is a homogeneous d -graph. This is illustrated by the following three examples.

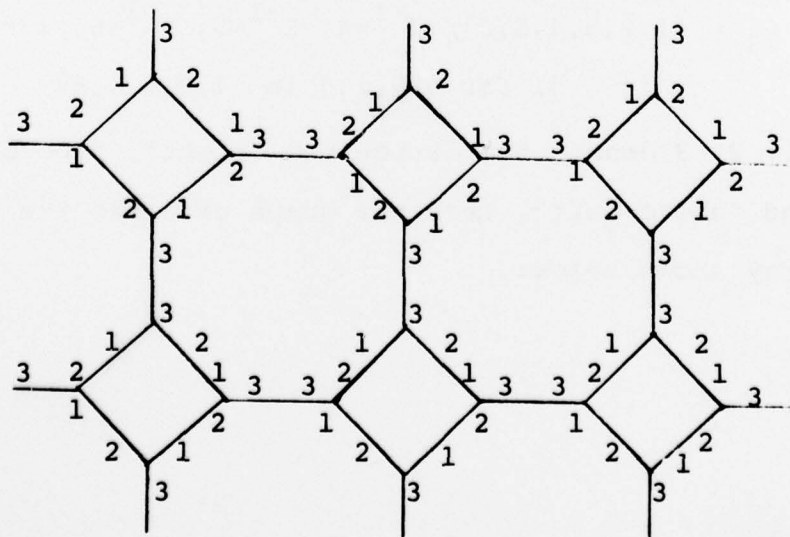
(1) Hexagonal arrays and (semi)-regular tessellations

$$\text{Let } G_1 = \{1, 2, 3, 4, 5, 6\} / \{1^{-1}=4, 2^{-1}=5, 3^{-1}=6, 13=2, \\ ij = ji \text{ for any } i, j \text{ in } 1, 2, \dots, 6\}$$

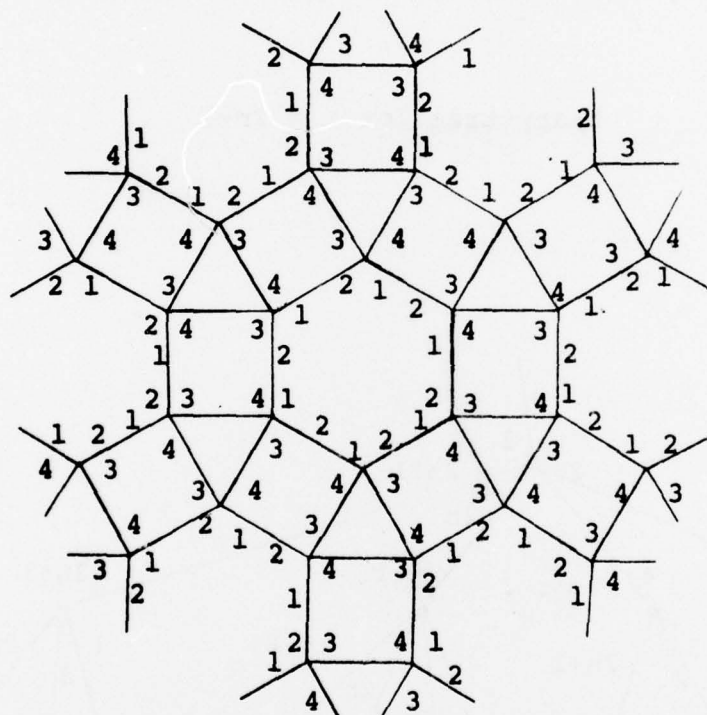
If 1, 2, 3 denote the directions "right", "above right", and "above left", then the graph of G_1 is the hexagonal array shown below:



The three regular tessellations and eight semi-regular (Archimedean) tessellations in the Euclidean plane as shown on pages 24, 41 and 42 of [7] can all be regarded as homogeneous d-graphs. For example, the tessellation $(4,8,8)$, which has three polygonal faces surrounding each vertex, where the numbers of sides of the faces are 4, 8 and 8, can be represented by the generators $\{1,2,3\}$ and the relations $1^{-1}=2$, $3^{-1}=3$, $1111=e$, $13131313=e$ if the arc ends are numbered as follows:



The more complex tessellation $(3,4,6,4)$, which has four polygonal faces surrounding each vertex, where the numbers of sides of the faces are 3, 4, 6, and 4 in cyclic order, can be represented by the generators $\{1, 2, 3, 4\}$ and the relations $\{1^{-1}=2, 3^{-1}=4, 333=e, 1313=3, 1^6=e\}$, if the arc ends are numbered as



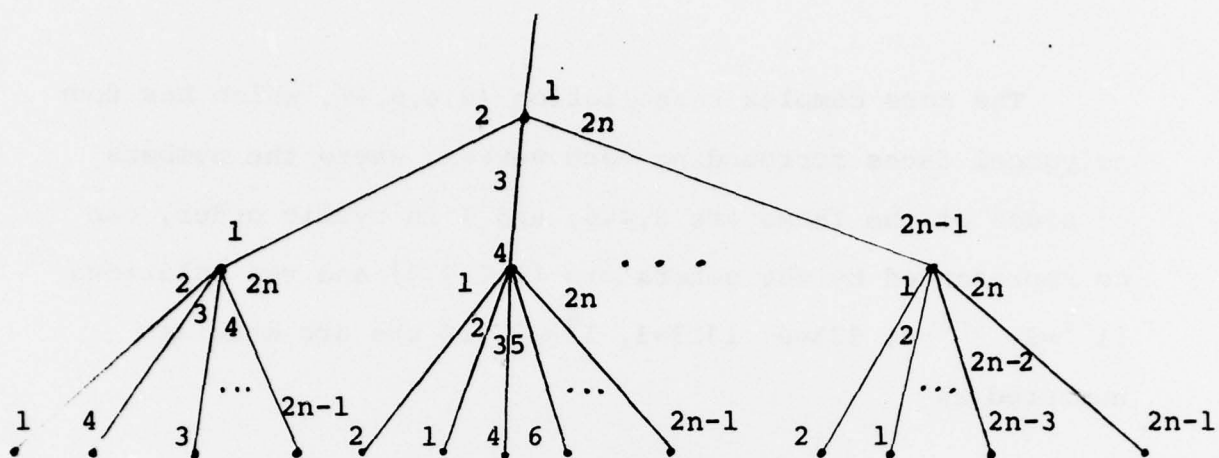
(2) t-ary trees

In a t -ary tree each node has $t+1$ neighbors. For $n \geq 1$ let $G_{2,n} = \{1, \dots, 2n\} / \{i^{-1} = i-1 \mid i=2, 4, \dots, 2n\}$

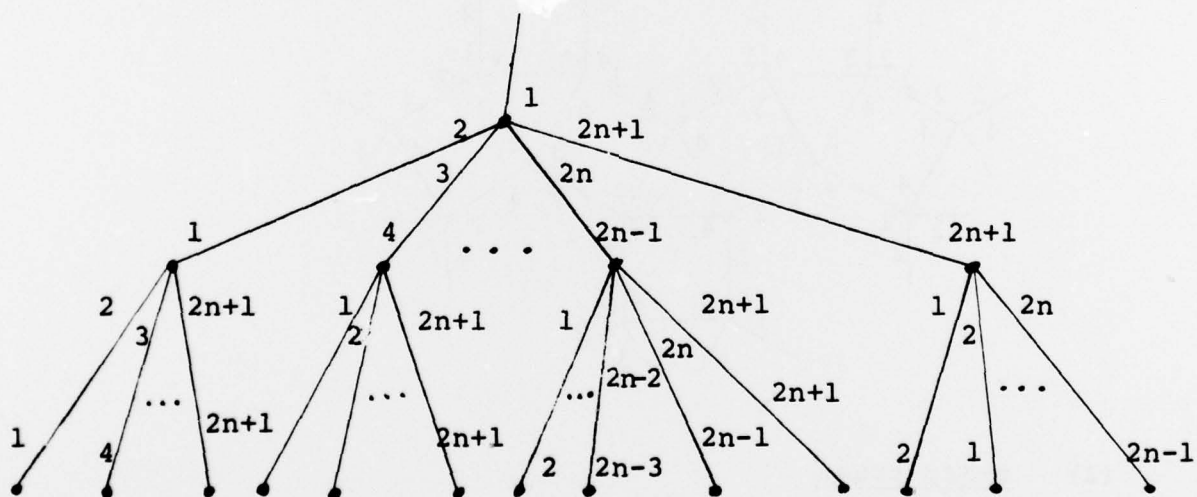
$$G'_{2,n} = \{1, \dots, 2n+1\} / (\{i^{-1} = i-1 \mid i=2, 4, \dots, 2n\} \cup \{(2n+1)^{-1} = 2n+1\})$$

Then the graph of $G_{2,n}$ is a t -ary tree for $t = 2n-1$,

and the graph of $G'_{2,n}$ is a t -ary tree for $t = 2n$.



t -ary tree for $t = 2n-1$



t -ary tree for $t = 2n$

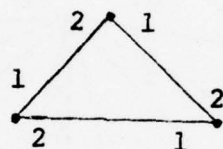
Note that G_2 and G'_2 are non-abelian since $23 \neq 32$ in both groups.

(3) Complete graphs

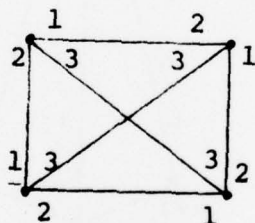
For $n \geq 1$, let $D_{2n+1} = \{i^{-1} = n+i \mid 1 \leq i \leq n\} \cup \{1^j_j = e \mid 2 \leq j \leq n\} \cup \{1^{2n+1} = e\}$;
 $D_{2n+2} = \{i^{-1} = n+i \mid 1 \leq i \leq n\} \cup \{(2n+1)^{-1} = 2n+1, 1^{2n+2} = e, 1^{n+1}(2n+1) = e\}$
 $\cup \{1^j_j \mid 2 \leq j \leq n\}$.

It is not hard to show that the graph of $G_i = \{1, \dots, i-1\}/D_i$, $i \geq 3$ is C_i , the complete graph with i nodes.

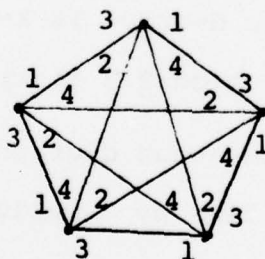
C_3 :



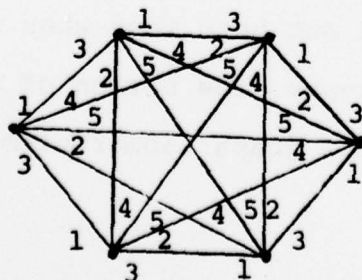
C_4 :



C_5 :

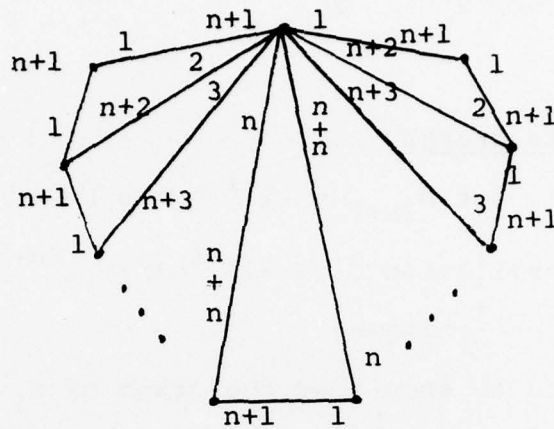


C_6 :

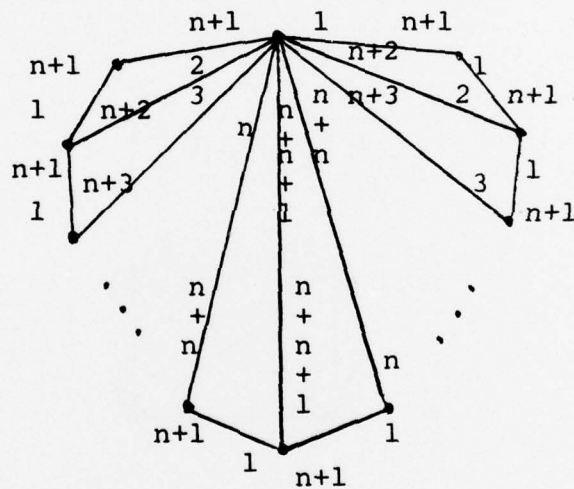


In general:

C_{2n+1} :



C_{2n+2} :



Since every homogeneous d-graph is k-locally-homogeneous for all k, the results of Section 2.3 imply

Proposition 9. For any homogeneous d-graph, subgraph matching can be done in diameter time by a cellular d-graph automaton.

It should be pointed out here that when we consider arrays as homogeneous d-graphs, the notion of direction in an array is not important in graph isomorphism, namely

$\begin{pmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{pmatrix}$ is isomorphic to $\begin{pmatrix} \cdot & \cdot \\ \cdot & \cdot \end{pmatrix}$ and $\begin{pmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{pmatrix}$ and $\begin{pmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{pmatrix}$.

3.5 Application: clique finding in d-graphs

In a clique all the nodes are neighbors of each other. Since every node of a d-graph has at most d non-# neighbors, the size of any clique in a d-graph is $\leq d+1$. This makes the clique finding problem in a d-graph much easier than that in a general graph.

Let us first consider the simple problem of finding the largest clique that a given node n belongs to. At the first step, each non-# ith neighbor of n marks its state with i ($1 \leq i \leq d$). At the next step, each neighbor m of n writes in its state a list i_1, \dots, i_t if the i_1 th, \dots , i_t th neighbors of n are also neighbors of m (the list may be empty). Thus at the third step, n can tell from its neighbors' states which neighbors form cliques with it, and the sizes of the cliques. It is easy for n to record in its state the size of a largest clique and the numbers of the neighbors which are nodes of the largest clique. It is also not hard to mark the largest clique or even to mark all the cliques that n belongs to, because the number of such cliques is $\leq \binom{d}{1} + \binom{d}{2} + \dots + \binom{d}{d} < 2^d$. Therefore the time required to find the cliques at a node is constant.

Now consider the problem of finding the size of a maximal clique in a d-graph Γ in diameter (of Γ) time. If we find the size of the largest clique at each node of Γ , one node at a time, and then transmit the maximum of the sizes to the distinguished node, this takes area time. But largest clique finding at many nodes simultaneously will

involve difficulties, since the signals from different nodes are not distinguishable. A better approach is to try to find subgraphs of Γ isomorphic to C_{d+1} , i.e., cliques of size $d+1$; if none exist, then we try subgraphs isomorphic to $C_d, C_{d-1}, \dots, C_3, C_2$ in order (there are always subgraphs isomorphic to C_2 if Γ is connected and has more than one non-# node). When for some i , a subgraph isomorphic to C_i is found, i is transmitted to the distinguished node D as the size of the maximal clique in Γ . When attempting to find subgraphs isomorphic to C_i ($2 \leq i \leq d+1$) in diameter time, the difficulties of subgraph matching in a general d -graph as discussed in Section 2.1 also arise. However, if Γ is homogeneous, or 3-locally-homogeneous or 1-level-colored, we can detect the existence or nonexistence of C_i in diameter (of Γ) time. Therefore the size of a maximal clique can be transmitted and recorded in the state of the distinguished node of Γ in time proportional to the diameter of Γ , since there are at most d C_i 's to be checked.

When a subgraph isomorphic to C_i is identified, the node n of Γ corresponding to a special node (say A , the root of a spanning tree T_{C_i}) of C_i can be identified and the subgraph isomorphic to C_i can be recorded in n 's state. Therefore it is easy to mark the cliques of Γ isomorphic to C_i . Note that if the nodes of C_i have the same label, then when node n identifies itself as corresponding to node A of C_i , there are $(i-1)!$ different correspondences of C_i to the same $i-1$ neighbors of n in Γ , since each qualifies as

any one of the $i-1$ nodes of C_i . n can get rid of these redundant assignments by just specifying which $i-1$ of its neighbors belong to C_i . It is also straightforward to see that each node can record in its state the size of the largest clique it belongs to and which of its neighbors form such largest cliques.

4. Concluding remarks

Diameter time algorithms for the graph and subgraph matching problems are presented for trees, k -level-colored d -graphs, k -locally-homogeneous d -graphs, and homogeneous d -graphs. If fast algorithms are found to k -level-color a d -graph, then we will also have a fast algorithm for subgraph matching.

k -local-homogeneity seems somewhat artificial; however, a special case of it, namely homogeneity, holds for many important classes of d -graphs.

Homogeneous d -graphs may be considered as a natural generalization of both arrays and trees. The arc end numbering of a homogeneous d -graph is consistent. The description of the homogeneity conditions is the same at each node. In general, the description is also finite and compact (for example using group presentation), so that it can easily be stored in the finite state automaton at each node of the d -graph. It would be of interest to further study homogeneous d -graphs.

References

1. A. Wu, Cellular graph acceptors, TR-599, Computer Science Center, University of Maryland, College Park, MD, November 1977.
2. A. Wu, Cellular acceptors, 2, TR-621, Computer Science Center, University of Maryland, College Park, MD, December 1977.
3. A. N. Shah, D. L. Milgram, and A. Rosenfeld, Parallel web automata, TR-231, Computer Science Center, University of Maryland, College Park, MD, February 1973.
4. A. Rosenfeld, Networks of automata -- some applications, IEEE Trans. SMC-5, 1975, 380-383.
5. J. P. Mylopoulos and T. Pavlidis, On the topological properties of quantized spaces, I. The notion of dimension, J.ACM, April 1975, 239-246.
6. J. P. Mylopoulos, On the definition and recognition of patterns in discrete spaces, Princeton University Computer Science Laboratory Technical Report 84, August 1970.
7. L. Fejes Toth, Regular Figures, Macmillan, NY, 1964.

4050360

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

18 (19) REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR-78-1007	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) CELLULAR GRAPH ACCEPTORS, 3.	5. TYPE OF REPORT & PERIOD COVERED 9 Interim rept.	
7. AUTHOR(s) 10 Angela Wu	8. CONTRACT OR GRANT NUMBER(s) 15 AFOSR-77-3271	
9. PERFORMING ORGANIZATION NAME AND ADDRESS University of Maryland Computer Science Center College Park, MD 20742	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102F 2304/A2	
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Office of Scientific Research/NM Bolling AFB, Washington, DC 20332	12. REPORT DATE 11 Apr 78	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 14 TR-648	13. NUMBER OF PAGES 43	
	15. SECURITY CLASS. (of this report) UNCLASSIFIED 12/46p.	
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE		
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Cellular automata Graph automata Graph isomorphism		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) In earlier reports, cellular acceptors were studied whose languages are sets of d-graphs, i.e., labelled graphs of bounded degree whose arcs at each node are numbered. This report discusses acceptance tasks that depend on the concept of d-graph isomorphism -- in particular, the task of deciding whether a d-graph has a d-subgraph isomorphic to a given d-graph.		

DD FORM 1 JAN 73 1473

EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

403018